

COMPUTATIONS OF GENERATING LENGTHS WITH GAP

GAOHONG WANG

ABSTRACT. In this paper, we discuss how to apply GAP to do computations in modular representation theory. Of particular interest is the generating number of a group algebra, which measures the failure of the generating hypothesis in the stable module category. We introduce a computational method to do this calculation and present it in pseudo-code. We have also implemented the algorithm in GAP and managed to do computations of examples that we were not able to do before. The computations lead to conjectures on the ghost numbers of the groups Q_8 and A_4 .

CONTENTS

1. Introduction	1
2. Background	3
2.1. The stable module category	3
2.2. The generalised generating hypothesis and projective classes	4
3. A computational method to calculate the generating length	5
4. New algorithms developed for computations in $\mathbf{StMod}(kG)$	7
4.1. The ReplaceWithInj function and the Simple function	7
4.2. Other functions related to ReplaceWithInj	10
4.3. The ProjectiveFreeSummand function	10
5. Examples	11
5.1. Comparing the new code with the old code	11
5.2. Computations in C_9 , Q_8 , and A_4	12
5.3. The group $C_3 \times S_3$ at the prime 3	13
References	15

1. INTRODUCTION

In this paper, we develop new algorithms to do computations in modular representation theory, and present them in pseudo-code. We have implemented the code in GAP [13], which is a system for computational discrete algebra, building upon the GAP package ‘reps’ developed by Webb and others [9], and some extra functions written by Christensen that supplement those in the main file. The code allows us to do computations of examples that we were not able to do before. And the computations lead to some conjectures in these examples.

Let G be a finite group, and let k be a field whose characteristic divides the order of G . We are interested in the generating number of the group algebra kG , which is a numeric invariant of the stable module category $\mathbf{StMod}(kG)$ that measures the failure of the generating hypothesis. We will provide more background on the stable module category and the generating number in Section 2. Briefly speaking, the stable module category $\mathbf{StMod}(kG)$ is a quotient category of the module category $\mathbf{Mod}(kG)$, where the projective modules are killed. The stable module category is a triangulated category, so we can study the generalised generating hypothesis in $\mathbf{StMod}(kG)$. This is motivated by the famous conjecture by Peter Freyd in stable homotopy theory, which states that if a map between two compact spectra is sent to zero by the stable homotopy group functor, then

the map is null homotopic. The conjecture is referred to as the generating hypothesis and is still an open question. Generalising to a triangulated category \mathbf{T} together with a set of distinguished objects \mathbb{S} , the set of graded functors $[S, -]_*$ with $S \in \mathbb{S}$ is analogous to the stable homotopy group functor in the sense that

$$\text{if } [S, M]_* = 0 \text{ for all } S \in \mathbb{S} \text{ and } M \in \text{Loc}\langle \mathbb{S} \rangle, \text{ then } M = 0.$$

Here $[-, -]_*$ denotes the graded hom-sets in \mathbf{T} . We say that \mathbf{T} satisfies **the generating hypothesis with respect to \mathbb{S}** if the functors $[S, -]_*$ are faithful on $\text{Thick}\langle \mathbb{S} \rangle$ for all $S \in \mathbb{S}$. See Section 2.2 for the definition of $\text{Thick}\langle \mathbb{S} \rangle$ and more details.

The generalised generating hypothesis has been studied in various cases, such as the derived category of a ring R and the stable module category of a group algebra kG . For the stable module category, we take $\mathbb{S} = \{k\}$, hence $[k, -]_* \cong \widehat{H}^*(G, -)$ is the Tate cohomology. It is known that the generating hypothesis fails in $\text{StMod}(kG)$ for most groups [2, 4, 5, 7]. In this case, we can study the degree to which the generating hypothesis fails, and this is measured by the **generating number** of the group algebra. We call a map in the kernel of Tate cohomology a **ghost**. Roughly speaking, we consider the n -fold composite of ghosts out of a module M , and the smallest integer n such that each such composite is stably trivial provides an invariant of M called the **generating length** of M . The **generating number** of kG is defined to be the least upper bound of the generating lengths of modules in $\text{Thick}\langle k \rangle$, and one can show that the generating hypothesis holds in $\text{StMod}(kG)$ if and only if the generating number of kG is 1 [4]. This idea is formalised in a projective class, which we discuss in Section 2.2. We also show that there are equivalent characterisations of the generating length. For example, we can consider the n -fold composite of *universal* ghosts out of the module M , and the generating length of M is the smallest integer n such that the n -fold composite of universal ghosts out of M is stably trivial. See Section 2.2 for more details on universal ghosts and generating lengths.

In Section 3, we show how the idea of universal ghosts can be applied to compute the generating length. In general, computing the universal ghost involves modules of infinite dimension. We prove that the generating length is the limit of a sequence of *unstable* lengths (Corollary 3.4). The unstable lengths are computable using only modules of finite dimension.

In Section 4, we introduce an algorithm for replacing a map with an injection, which is essential for the computation of the universal ghost. More precisely, we can replace a map with an injection by adding a projective summand to the codomain of the map. Since projective modules are isomorphic to zero in $\text{StMod}(kG)$, this replacement is equivalent to the original map. The existing code in the extra functions computes the replacement by adding a free module to the codomain. As a consequence, the cokernel of the replacement can contain projective summands. We introduce a new algorithm to do this computation and implement it in GAP [15]. See Section 5 for examples showing that the new method is faster. The idea is to first replace the free module by a direct sum of indecomposable projective modules. Then we prove a condition that determines whether we need to add a map $g : M \rightarrow P$ to the original map $f : M \rightarrow N$. More precisely, we will replace f by $f + g : M \rightarrow N \oplus P$ if the following condition is satisfied:

$$\ker(\text{Hom}(S, f + g)) \subsetneq \ker(\text{Hom}(S, f)),$$

where $S = P/\text{rad}(P)$ is the simple module corresponding to the indecomposable projective module P . Roughly speaking, we are using the fact that the map f is injective if and only if it is injective on the socle. This can be determined by a rank computation and is presented in pseudo-code in Section 4.1. We also show that the method provides an optimal answer in the sense that the replacement is minimal. This new function `ReplaceWithInj`, together with the function that computes the (unstable) universal ghost, allows us to compute the generating length of a module within a finite range. We present some other functions related to `ReplaceWithInj` in Section 4 as well. For example, we need to compute the `Simple` module when we check the condition displayed above. And we have a dual function `ReplaceWithSurj` that replaces a map with a surjection. We also introduce

a new algorithm to compute the projective-free summand of a module in Section 4.3, and show that the idea in `ReplaceWithSurj` can be applied to improve the algorithm.

The generating number of a group algebra is studied for a p -group in [6, 10] and for a non- p -group in [11], where theoretical and computational results are given for generating numbers and their bounds. In particular, we know that the generating number of kG is finite, provided that $\text{Thick}\langle k \rangle$ is contained in the principal block B_0 . But when the condition $\text{Thick}\langle k \rangle = \text{StMod}(B_0)$ fails, we know of no examples where we can compute the generating number or an upper bound. It is sometimes not easy to determine whether a single module is contained in $\text{Thick}\langle k \rangle$. For example, we consider the group $C_3 \times S_3$ over a field k of characteristic 3 in Section 5.3, where we compute the generating length of a $k(C_3 \times S_3)$ -module and show that it is in $\text{Thick}\langle k \rangle$. We also make computations for the groups Q_8 and A_4 in Section 5.2, providing evidence for the conjectures that the generating number of Q_8 is 3 and that the generating number of A_4 is 2.

We give a brief summary of the contents of the paper to end the introduction: In Section 2, we provide background material for the stable module category and the generalised generating hypothesis, and define the generating number of a group algebra. In Section 3, we introduce an algorithm for computing the unstable length of a module within a finite range and prove that the generating length of the module is the limit of the unstable lengths as the range goes to infinity. In Section 4, we describe a new algorithm for the function `ReplaceWithInj` that replaces a map with an injection and introduce other related functions. In Section 5, we present examples of computations with the new code, showing that the new code is faster, as well as providing evidence for the conjectures on the generating numbers of the groups Q_8 and A_4 .

2. BACKGROUND

In this section, we review some background on modular representation theory and introduce some general concepts that will be needed in the rest of the paper.

2.1. The stable module category

Let G be a finite group and k be a field whose characteristic divides the order of G . The stable module category $\text{StMod}(kG)$ is a quotient category of the module category $\text{Mod}(kG)$. The hom-set between two modules M and N in $\text{StMod}(kG)$ is defined by

$$\underline{\text{Hom}}(M, N) := \text{Hom}(M, N) / \text{PHom}(M, N),$$

where $\text{PHom}(M, N)$ consists of stably-trivial maps between M and N , i.e., the maps that factor through a projective module P . Note that projective modules are isomorphic to zero in the stable module category. To avoid ambiguity, we say that two modules M and N are **stably isomorphic** if they are isomorphic in $\text{StMod}(kG)$. We write $\text{stmod}(kG)$ for the full subcategory of all finite-dimension modules in $\text{StMod}(kG)$. Then $\text{stmod}(kG)$ consists of exactly the **compact** objects M in $\text{StMod}(kG)$ such that the canonical map

$$\oplus \underline{\text{Hom}}(M, X_i) \rightarrow \underline{\text{Hom}}(M, \oplus X_i)$$

is an isomorphism for any class of objects $\{X_i\}$ in $\text{StMod}(kG)$. Since the regular representation kG is both projective and injective as a module over itself, projective and injective modules coincide in $\text{stmod}(kG)$. It also follows that two modules are stably isomorphic in $\text{stmod}(kG)$ if and only if they have isomorphic projective-free summands.

The stable module category has a triangulation structure which we now describe. Then one will see that cohomology groups of kG -modules are represented by hom-sets in the stable module category. The desuspension ΩM of a module $M \in \text{StMod}(kG)$ is defined to be the kernel of a surjective map $P \rightarrow M$, where P is a projective module. Note that ΩM is well defined up to isomorphism in $\text{StMod}(kG)$. We write $\Omega^n M$ for the n -fold desuspension of M . Dually, we can define ΣM by the short exact sequence $0 \rightarrow M \rightarrow P \rightarrow \Sigma M \rightarrow 0$, where P is a projective (and injective) module. Now we define the group cohomology and Tate cohomology of a kG -module M .

Definition 2.1. Let G be a finite group and k be a field. Let

$$P_* : \cdots \longrightarrow P_2 \longrightarrow P_1 \longrightarrow P_0$$

be a projective resolution of the trivial representation k . The n -th **group cohomology** $H^n(G, M)$ of M is defined to be the n -th cohomology of the chain complex $\text{Hom}(P_*, M)$ for $n \geq 0$.

If, instead of a projective resolution, we take a complete resolution

$$T_* : \cdots \longrightarrow P_1 \longrightarrow P_0 \xrightarrow{\partial_0} P_{-1} \longrightarrow P_{-2} \longrightarrow \cdots$$

of k , that is, a doubly infinite exact sequence of projective modules such that $\text{im}(\partial_0) = k$, then the n -th **Tate cohomology** $\widehat{H}^n(G, M)$ of M is defined to be the n -th cohomology of the chain complex $\text{Hom}(T_*, M)$ for $n \in \mathbb{Z}$. We can also replace the trivial module k by an arbitrary kG -module L and compute the resolutions P_* and T_* of L . The cohomology of the chain complexes $\text{Hom}(P_*, M)$ and $\text{Hom}(T_*, M)$ of M are denoted by $\text{Ext}^n(L, M)$ and $\widehat{\text{Ext}}^n(L, M)$, respectively.

It is easy to see that, for M and L in $\text{StMod}(kG)$ and $n \in \mathbb{Z}$, there is a natural isomorphism

$$\widehat{\text{Ext}}^n(L, M) \cong \underline{\text{Hom}}(L, \Sigma^n M) \cong \underline{\text{Hom}}(\Omega^n L, M).$$

In particular, the Tate cohomology $\widehat{H}^n(G, M)$ of M is represented by the trivial representation k as $\underline{\text{Hom}}(\Omega^n k, M)$. Moreover, by usual homological algebra, $\widehat{\text{Ext}}^1(L, M)$ is equivalent to the isomorphism classes of extensions between L and M . Hence a short exact sequence $0 \rightarrow M \rightarrow N \rightarrow L \rightarrow 0$ corresponds to a map $\delta \in \underline{\text{Hom}}(L, \Sigma M)$. This defines a triangle in $\text{StMod}(kG)$:

$$M \rightarrow N \rightarrow L \xrightarrow{\delta} \Sigma M,$$

and gives $\text{StMod}(kG)$ a triangulation. To compute the **cofibre** of a map $f : M \rightarrow N$, we need to replace f with an injection that is stably equivalent to it. For simplicity, we write $f + g$ for the map $M \rightarrow N \oplus P$, where $f : M \rightarrow N$ and $g : M \rightarrow P$ are maps out of M . If P is projective, then the maps f and $f + g$ are stably equivalent. Choosing a map $g : M \rightarrow P$ such that $f + g$ is injective, then the cofibre of f in $\text{StMod}(kG)$ is defined to be the cokernel of $f + g$. Note again that the cofibre is well-defined up to isomorphism in $\text{StMod}(kG)$. Dually, we can define the fibre of a map f . In Section 4, we will present the pseudo code to compute the replacement of a map with an injection.

2.2. The generalised generating hypothesis and projective classes

In this section, we introduce the generalised generating hypothesis in a triangulated category, and discuss its relation with a projective class. Then we show how to apply this idea to $\text{StMod}(kG)$.

Let \mathbf{T} be a triangulated category, and let \mathbb{S} be a set of distinguished objects in \mathbf{T} . We write $[-, -]$ for hom-sets in \mathbf{T} . A full subcategory \mathbf{S} of \mathbf{T} is said to be **thick** if it is closed under suspension, desuspension, retracts, and triangles. If in addition, \mathbf{S} is closed under arbitrary sums, then it is called a **localising** subcategory of \mathbf{T} . The thick (resp. localising) subcategory generated by \mathbb{S} is the smallest thick (resp. localising) subcategory that contains \mathbb{S} , and is denoted by $\text{Thick}\langle\mathbb{S}\rangle$ (resp. $\text{Loc}\langle\mathbb{S}\rangle$). The set of graded functors $[S, -]_*$ with $S \in \mathbb{S}$ is analogous to the stable homotopy group functor in the sense that

$$\text{if } [S, M]_* = 0 \text{ for all } S \in \mathbb{S} \text{ and } M \in \text{Loc}\langle\mathbb{S}\rangle, \text{ then } M = 0.$$

But in general, we don't expect that $[S, -]_*$ is faithful when restricted to $\text{Loc}\langle\mathbb{S}\rangle$, or, in other words, $[S, -]_*$ will detect not only zero objects, but also zero maps. However, we can restrict the functors $[S, -]_*$ further to $\text{Thick}\langle\mathbb{S}\rangle$. We say that \mathbf{T} satisfies **the generating hypothesis with respect to** \mathbb{S} if the functors $[S, -]_*$ are faithful on $\text{Thick}\langle\mathbb{S}\rangle$ for all $S \in \mathbb{S}$. Note that if \mathbb{S} consists of finitely many compact objects in \mathbf{T} , then

$$\text{Thick}\langle\mathbb{S}\rangle = \text{Loc}\langle\mathbb{S}\rangle \cap \text{compact objects in } \mathbf{T}.$$

We write \mathcal{I} for the intersection of the kernels of $[S, -]_*$ for all $S \in \mathbb{S}$. If \mathcal{I} is the zero ideal, then the generating hypothesis holds. Note that this is a stronger condition than the generating hypothesis.

Nevertheless, when the generating hypothesis fails, the least integer n such that \mathcal{I}^n is zero provides some measurement of the failure of the generating hypothesis, where \mathcal{I}^n is the n -th power of the ideal \mathcal{I} that consists of composites of n -fold maps in \mathcal{I} . We formalise this idea in the concept of a projective class:

Definition 2.2. Let T be a triangulated category. A **projective class** in T consists of a class \mathcal{P} of objects of T and an ideal \mathcal{I} of morphisms of T such that:

- (i) \mathcal{P} consists of exactly the objects P such that every composite $P \rightarrow X \rightarrow Y$ is zero for each $X \rightarrow Y$ in \mathcal{I} ,
- (ii) \mathcal{I} consists of exactly the maps $X \rightarrow Y$ such that every composite $P \rightarrow X \rightarrow Y$ is zero for each P in \mathcal{P} .
- (iii) for each X in T , there is a triangle $P \rightarrow X \rightarrow Y \rightarrow \Sigma P$ with P in \mathcal{P} and $X \rightarrow Y$ in \mathcal{I} .

Note that \mathcal{P} is closed under retracts and arbitrary direct sums. If \mathcal{P} (or equivalently \mathcal{I}) is closed under suspension and desuspension, then we say that the projective class $(\mathcal{P}, \mathcal{I})$ is **stable**. The map $X \rightarrow Y$ in the third condition is a universal map out of X in \mathcal{I} . In general, for a class of objects \mathcal{P} , we can define a nested sequence of classes by

- (i) $\mathcal{P}_1 = \mathcal{P}$, and
- (ii) $X \in \mathcal{P}_n$ if X is a retract of some object M such that M sits in a triangle $P \rightarrow M \rightarrow Q$ with $P \in \mathcal{P}$ and $Q \in \mathcal{P}_{n-1}$.

For an object X in T , the **length** $\text{len}(X)$ of X with respect to $(\mathcal{P}, \mathcal{I})$ is defined to be the smallest integer n such that $X \in \mathcal{P}_n$, if such an n exists. There is an alternative interpretation of $\text{len}(X)$ using \mathcal{I}^n by the property of a projective class, which we state as the next lemma. By convention, \mathcal{P}_0 consists of all zero objects in T and \mathcal{I}^0 consists of all maps in T .

Lemma 2.3 ([8]). *Let T be a triangulated category, and $(\mathcal{P}, \mathcal{I})$ be a (possibly unstable) projective class in T . Then, for all integers $n \geq 0$, $(\mathcal{P}_n, \mathcal{I}^n)$ is a projective class in T . In particular, the following conditions are equivalent for an object X in T :*

- (i) X is in \mathcal{P}_n .
- (ii) Every n -fold composite of maps in \mathcal{I} out of X is zero.
- (iii) The n -fold composite of universal maps in \mathcal{I} out of X is zero.

Now we consider $\text{StMod}(kG)$ and the Tate cohomology functor. We call a map in the kernel of the Tate cohomology functor a **ghost** and write \mathcal{G} for the ideal of ghosts in $\text{StMod}(kG)$. Let \mathcal{F} be the class of objects in $\text{StMod}(kG)$ generated by the trivial representation k under retracts, arbitrary direct sums, suspension and desuspension. Since the Tate cohomology is represented by k , the pair $(\mathcal{F}, \mathcal{G})$ forms a projective class in $\text{StMod}(kG)$, and this is called the **ghost projective class**. For $M \in \text{Thick}(k)$, the **generating length** $\text{gel}(M)$ of M is the length of M with respect to $(\mathcal{F}, \mathcal{G})$. The generating number of kG is defined to be the least upper bound of $\text{gel}(M)$ for all $M \in \text{Thick}(k)$.

There is another invariant called the *ghost number* that is more closely related to the generating hypothesis in $\text{StMod}(kG)$. In general, the ghost number of kG is less than or equal to the generating number. But in the examples that we are able to compute, we have shown them to be equal. We will focus on the computation of the generating number in this paper. See [6, 10, 11] for further discussions on the difference between the ghost number and the generating number.

3. A COMPUTATIONAL METHOD TO CALCULATE THE GENERATING LENGTH

By Lemma 2.3, the generating length of a module M can be computed using universal ghosts. The idea is presented in the following pseudo-code:

```

LengthHelper = function with inputs:
    a map f from M to N and an integer n
    g = universal ghost from N to L

```

```

if f composed with g is stably trivial then
  return f and n
return LengthHelper(f composed with g, n+1)

```

Length of M = LengthHelper(the identity map on M , 1)

However, computing the universal ghost involves modules of infinite dimension, unless the Tate cohomology of M is finitely generated. Hence, to make this idea work, we need to first consider unstable ghosts within a finite range. Let $\mathcal{F}(-m, m)$ be the class generated by $\{\Sigma^i k \mid -m \leq i \leq m\} \subseteq \mathcal{F}$ under retracts and arbitrary direct sums. Then $\mathcal{F}(-m, m)$ forms part of a projective class, and the relative null maps consists of the unstable ghosts within the range $[-m, m]$. Given $M \in \text{Thick}\langle k \rangle$, we write $\text{gel}_m(M)$ for the length of M with respect to $\mathcal{F}(-m, m)$. Since $\mathcal{F}(-m, m) \subseteq \mathcal{F}(-m-1, m+1) \subseteq \cdots \subseteq \mathcal{F}$, we get a decreasing sequence greater than or equal to $\text{gel}(M)$:

$$\text{gel}_m(M) \geq \text{gel}_{m+1}(M) \geq \cdots \geq \text{gel}(M),$$

where $\text{gel}_m(M)$ can be computed using the pseudo-code presented above for each $m \geq 0$.

Example 3.1. Let G be a p -group and let k be a field of characteristic p . Let M be a projective-free kG -module. Then $\text{gel}_0(M)$ is equal to the radical length of M [10, Proposition 4.5].

Example 3.2. If the cohomology of kG has periodicity n , then $\text{gel}(M) = \text{gel}_{\lfloor \frac{n}{2} \rfloor}(M)$ for all $M \in \text{Thick}\langle k \rangle$, and the computation of the generating length of M is a finite process.

Now we want to show that the limit of $\text{gel}_m(M)$ is $\text{gel}(M)$. This will be a corollary of Lemma 3.3. We need some more notations before we introduce the lemma. Let \mathcal{T} be a triangulated category with compact objects \mathcal{T}^c . Let \mathbb{P} be a set of compact objects in \mathcal{T} . We write \mathcal{P} for the class of objects generated by \mathbb{P} under retracts, arbitrary direct sums, suspension, and desuspension and write \mathcal{P}^c for the class of objects generated by \mathbb{P} under retracts, finite sums, suspension, and desuspension. Note that $\mathcal{P}^c = \mathcal{P} \cap \mathcal{T}^c$. More generally, we can define $\mathcal{P}_n^c := (\mathcal{P}^c)_n$ in a similarly pattern as \mathcal{P}_n , and the following lemma holds:

Lemma 3.3 ([3, Proposition 2.2.4]). *Let \mathcal{T} be a triangulated category, and let \mathbb{P} be a set of compact objects in \mathcal{T} . With the notation introduced above,*

$$\mathcal{P}_n^c = \mathcal{P}_n \cap \mathcal{T}^c.$$

In particular, $\text{Thick}\langle \mathbb{P} \rangle = \text{Loc}\langle \mathbb{P} \rangle \cap \mathcal{T}^c$. Moreover, the sequence

$$\mathcal{P}_1^c \subseteq \mathcal{P}_2^c \subseteq \cdots \subseteq \mathcal{P}_n^c \subseteq \cdots \subseteq \text{Thick}\langle \mathbb{P} \rangle$$

is a filtration of $\text{Thick}\langle \mathbb{P} \rangle$ with $\text{Thick}\langle \mathbb{P} \rangle = \bigcup \mathcal{P}_n^c$. □

As a corollary, we can compute the generating length in $\text{stmod}(kG)$.

Corollary 3.4. *Let G be a finite group and k be a field whose characteristic divides the order of G . Let M be a module in $\text{Thick}\langle k \rangle$. Then $\text{gel}(M) = \lim_{m \rightarrow \infty} \text{gel}_m(M)$.*

Since $(\text{gel}_m(M))$ is a sequence of integers, we conclude that $\text{gel}_m(M) = \text{gel}(M)$ for m large.

Proof. Consider $\mathbb{P} = \{k\}$ in $\text{StMod}(kG)$. Let M be a module in $\text{Thick}\langle k \rangle$. It follows from the lemma that $M \in \mathcal{P}_n^c$, with $n = \text{gel}(M)$. However, there are only finitely many spheres $\Sigma^{n_i} k$ needed to build up M in n steps. Hence there exists an integer m , such that $M \in (\mathcal{F}(-m, m))_n$, and $\text{gel}_m(M) \leq n = \text{gel}(M)$. Conversely, since $\mathcal{F}(-m, m)$ is contained in \mathcal{F} , $\text{gel}(M) \leq \text{gel}_m(M)$. □

Remark 3.5. We remark here that there is not a universal choice of N such that $\text{gel}_N(M) = \text{gel}(M)$ for all $M \in \text{Thick}\langle k \rangle$. Indeed, if the group cohomology is not periodic, then $\text{gel}_N(\Omega^n k) = \text{gel}(\Omega^n k)$ if and only if $N \geq |n|$, and the number N can be arbitrarily large. Note that the numbers $\text{gel}_n(M)$ give upper bounds of the generating length of M . Hence if a lower bound of the generating length of M is known, we can hope to get the exact answer of the generating length of M . It would also

be interesting to know whether there is a way to compute lower bounds for the generating length that converge to the correct answer.

4. NEW ALGORITHMS DEVELOPED FOR COMPUTATIONS IN $\text{StMod}(kG)$

We have improved the GAP code used in the ‘reps’ package [9] to compute the universal ghost and generating length. We introduce the function `ReplaceWithInj` in this section, which is essential for computing the universal ghost. We also show the relation of `ReplaceWithInj` with other functions.

4.1. The `ReplaceWithInj` function and the Simple function

Recall that the universal ghost is the cofibre of a map that is surjective on Tate cohomology, and computing the cofibre depends on a function that replaces a map with an injection that is stably equivalent to it. For simplicity, we write $f + g$ for the map $M \rightarrow N \oplus P$, where $f : M \rightarrow N$ and $g : M \rightarrow P$ are maps out of M . If P is projective, then the maps f and $f + g$ are stably equivalent. Now let $\{P_i\}$ be the set of non-isomorphic indecomposable projective kG -modules, and let \mathcal{B}_i be a basis for $\text{Hom}(M, P_i)$. Observe that the natural map

$$\alpha : M \rightarrow \bigoplus_i (\bigoplus_{g \in \mathcal{B}_i} P_i)$$

is injective. Then for any map $f : M \rightarrow N$, the map $f + \alpha$ is a replacement of f with an injection. But in this way, we will have added more maps than we need to the map f . For example, we don’t need the maps g with $\ker(f + g) = \ker(f)$. In fact, we can do better than this and get rid of more maps that we don’t want. We need a lemma before we state the condition that we will put on g .

Lemma 4.1. *Let $f : M \rightarrow N$ be a map in $\text{mod}(kG)$. Then the map f is injective if and only if, for any simple module S , the map*

$$\text{Hom}(S, f) : \text{Hom}(S, M) \rightarrow \text{Hom}(S, N)$$

is injective.

Proof. Since $\ker(\text{Hom}(S, f)) \cong \text{Hom}(S, \ker(f))$, the map f being injective implies that $\text{Hom}(S, f)$ is injective for any $S \in \text{mod}(kG)$. Conversely, if $\text{Hom}(S, \ker(f)) = 0$ for all simple modules, then $\ker(f) = 0$ because the simple modules generate the module category. \square

It follows from the lemma that we only need to add to f those maps g that shrink $\ker(\text{Hom}(S, f))$ for some simple module S . Recall that, for each indecomposable projective module P , there is a simple module corresponding to it, given by $P/\text{rad}(P)$:

Lemma 4.2 ([1, Theorem 1.6.3]). *Let P be an indecomposable projective kG -module. Then the radical quotient $P/\text{rad}(P)$ is simple and $P/\text{rad}(P) \cong \text{soc}(P)$. Moreover, the assignment of $P/\text{rad}(P)$ to P provides a one-one correspondence between isomorphism classes of indecomposable projective kG -modules and simple kG -modules.* \square

Now let P be an indecomposable projective module, and let g be a map from M to P . We claim that, to decide whether we need to replace f by $f + g$, it suffices to check the condition

$$\ker(\text{Hom}(S, f + g)) \subsetneq \ker(\text{Hom}(S, f)), \quad (4.1)$$

for $S = P/\text{rad}(P)$. Indeed, if $S' \not\cong S$ is another simple module, then $\text{Hom}(S', P) = 0$, and since $\ker(\text{Hom}(S, f + g)) = \ker(\text{Hom}(S, f)) \cap \ker(\text{Hom}(S, g))$, there is no need to check g on S' .

It follows the discussion above that we can work with one indecomposable projective P at a time. Observe that if we have replaced f with $f' = f + g$, then we can replace the condition in Equation 4.1 with $\ker(\text{Hom}(S, f' + g)) \subsetneq \ker(\text{Hom}(S, f'))$. Also note that if $\{g_1, g_2, \dots, g_l\}$ is a basis for $\text{Hom}(M, P)$, then

$$\ker(\text{Hom}(S, \sum_{i=1}^l (g_i))) = \ker(\text{Hom}(S, \alpha)) = 0,$$

where $\alpha : M \rightarrow \bigoplus_i (\bigoplus_{g \in \mathcal{B}_i} P_i)$ is the injection we started with. Hence, the following pseudo-code produces a replacement f' of f such that $\ker(\text{Hom}(S, f')) = 0$ for $S = P/\text{rad}(P)$:

```

ReplaceWithInj = function with one input: a map f from M to N
  P = an indecomposable projective module
  S = the simple module corresponding to P
  for g in a basis for Hom(M, P)
    if ker(Hom(S, f+g)) is strictly contained in ker(Hom(S, f)) then
      replace f with f+g
    continue the loop of g until ker(Hom(S, f)) = 0
  return f

```

Then, by Lemma 4.1, we can loop the preceding process over all indecomposable projective modules and produce a replacement by an injection.

Remark 4.3. Note that the ‘for’ loop of g over $\text{Hom}(M, P)$ can be replaced any set of maps $\{g_1, g_2, \dots, g_l\}$ in $\text{Hom}(M, P)$, such that

$$\ker(\text{Hom}(S, \sum_{i=1}^l (g_i))) = 0.$$

In particular, if the injective hull of M has been computed, we can use it when we compute the replacement of a map $f : M \rightarrow N$ with an injection.

Now we describe how to check the condition whether $\ker(\text{Hom}(S, f + g)) \subsetneq \ker(\text{Hom}(S, f))$. This is done by a rank computation. We form the map $\beta : \bigoplus S \rightarrow M$, where the sum ranges over a basis for $\text{Hom}(S, M)$. Then we compare the dimensions of $\text{im}((f + g) \circ \beta)$ and $\text{im}(f \circ \beta)$ in the diagram

$$\begin{array}{ccccc}
 & & N \oplus P & & \\
 & \nearrow^{f+g} & & \searrow & \\
 \bigoplus S & \xrightarrow{\beta} & M & \xrightarrow{f} & N.
 \end{array}$$

It is clear that $\text{rank}((f + g) \circ \beta) \geq \text{rank}(f \circ \beta)$. Since $\bigoplus S$ is semi-simple, the equality holds if and only if $\ker(\text{Hom}(S, f + g)) = \ker(\text{Hom}(S, f))$. In other words, the following conditions are equivalent:

- (1) $\ker(\text{Hom}(S, f + g)) \subsetneq \ker(\text{Hom}(S, f))$,
- (2) $\text{rank}((f + g) \circ \beta) > \text{rank}(f \circ \beta)$.

Note that $\text{rank}(f \circ \beta)$ is at most $\text{rank}(\beta)$, and this is equivalent to $\ker(\text{Hom}(S, f)) = 0$, so we can break out the loop over the basis for $\text{Hom}(M, P)$ when $\text{rank}(f \circ \beta) = \text{rank}(\beta)$. We can also check at the same time whether f is injective or not and, if yes, we return f to avoid the extra loop over the other projective modules. To conclude the discussion, we display the function “ReplaceWithInj” in the following pseudo-code:

```

ReplaceWithInj = function with one input: a map f from M to N
  f = a given map from M to N
  if Rank(f) == dimension of M then    % f is injective
    return N and f
  L = list of non-isomorphic indecomposable projectives
  for P in L
    S = the simple module corresponding to P
    b = map from a sum of S to M, ranging over a basis for Hom(S, M)
    r = Rank(f composed with b)
    rankb = Rank(b)
    if r != rankb then
      % r not maximal, so need to loop over a basis for Hom(M, P)
      for g in a basis for Hom(M, P)

```



```

newf = f + g
newr = Rank(newf composed with b)
if newr > r then
    f = newf
    r = newr
    N = direct sum of N and P
if r == rankb then % r is maximal
    if Rank(f) == dimension of M then
        return N and f
    break out of the loop over the basis for Hom(M, P)
% This point should never be reached
return

```

Remark 4.4. The code produces an optimal answer in the sense that the replacement is minimal, unless the map f itself contains a stably trivial summand, in which case we need to exclude the summand. In particular, if N is the zero module, then we will compute the injective hull of M .

To see that the process is optimal, observe first that $\ker((f + g) \circ \beta) \subseteq \ker(f \circ \beta)$ is the kernel of the composite

$$\ker(f \circ \beta) \rightarrow \oplus S \xrightarrow{\beta} M \xrightarrow{g} P.$$

Since $\ker(f \circ \beta)$ is a direct sum of copies of the simple module S and P is the corresponding projective module, the image of this composite is either zero or isomorphic to S . It follows that, when we replace f by $f + g$, we always have

$$\text{rank}((f + g) \circ \beta) = \text{rank}(f \circ \beta) + \dim(S).$$

Thus, to replace a map $f : M \rightarrow N$ by an injection, we need to add exactly

$$\frac{\text{rank}(\beta) - \text{rank}(f \circ \beta)}{\dim(S)}$$

copies of the projective module P to N , as our code will do. Since this number is independent of the choice of a basis for $\text{Hom}(M, P)$, the code is optimal.

Note that the algorithm we introduced depends on a decomposition function to find all indecomposable projective modules and, for each indecomposable projective module, we need to find the corresponding simple module $S = P/\text{rad}(P)$.

To find $S = P/\text{rad}(P)$, observe that by Lemma 4.2, there is a self map on P

$$f : P \rightarrow P/\text{rad}(P) \cong \text{soc}(P) \rightarrow P,$$

with $\text{im}(f) \cong S$. Hence we can compute the image of *all* self maps on P to find S as the image whose dimension is the smallest, but this is not very efficient. So we replace P with $M = \text{im}(f)$, where f is a self map on P . Since M is both a submodule and a quotient module of P , it also satisfies the condition that $M/\text{rad}(M) \cong \text{soc}(M) \cong S$. Then we can find S as the image of a self map on M . To implement this idea, we can loop over all self maps f on P and compute $M = \text{im}(f)$. Then, if M is a proper submodule of P , we replace P with M and make a recursive call and compute the images of self maps on M . The recursion will end with a module S that has no proper submodules. In other words, S is simple. Note that if $\text{Hom}(M, M)$ has dimension 1 and $M/\text{rad}(M) \cong \text{soc}(M)$, then the map $M \rightarrow M/\text{rad}(M) \cong \text{soc}(M) \rightarrow M$ is an isomorphism, hence M is simple, and we can return M in this case. In conclusion, if P is an indecomposable projective module, then we can find the corresponding simple module S with the following pseudo-code:

```

Simple = a function with one input: a kG-module P such that
        P/rad(P) is isomorphic to soc(P)

```

```

hom = Hom(P, P)

```

```

if hom has dimension 1 then
    return P
for all maps f in hom
    if 0 < Rank(f) < dimension of P then
        return Simple(im(f))
% This point can be reached when k is not algebraically closed
return P

```

Remark 4.5. Note that not every simple module S has $\dim(\text{Hom}(M, M)) = 1$ when the field k is small. So, in general, we have to search over all self maps on M . Also note that, for an arbitrary module M , $\dim(\text{Hom}(M, M)) = 1$ does not imply that M is simple. For a counterexample, take $G = S_3$, the symmetric group on three letters, and consider the two dimensional module $M = \tilde{\Omega}k$, where the condition $M/\text{rad}(M) \cong \text{soc}(M)$ fails. However we have seen that the condition always holds for the module M that arises in this algorithm.

4.2. Other functions related to ReplaceWithInj

In this section, we show the relation of the function `ReplaceWithInj` with other functions.

(1) **Cofibre and Suspension.**

With the `ReplaceWithInj` function, we can compute the cofibre of a map f . In particular, replacing the zero map out of M , we get the injective hull of M , and its cofibre is the suspension of M . Since the `ReplaceWithInj` function provides an optimal answer, the suspension of M we get is projective-free. `Cofibre` is also essential in the `Length` function, where we need to compute universal ghosts.

(2) **CreateRandomModule.**

We can create random modules in $\text{Thick}\langle k \rangle$ using cofibres. We choose a random map $f : P \rightarrow Q$ between random modules P and Q that are sums of suspensions and desuspensions of k and compute the cofibre R_1 . Note that R_1 has generating length at most 2. Iterating the process n -times, we can build up a module R_n of length at most $n + 1$. Note that the function depends on the number of summands that we allow in each step and the number of steps n that we take.

(3) **IsStablyTrivial.**

Let $f : M \rightarrow P$ be an injection of M into a projective module. Then since P is also injective, every map from M to a projective module factors through f . Hence `ReplaceWithInj` provides an algorithm to detect whether a map $g : M \rightarrow N$ is stably-trivial or not, by checking whether it factors through f .

(4) **ReplaceWithSurj, Fibre and Desuspension.**

Since the pseudo-code we present in `ReplaceWithInj` is dualizable, we can write the dual functions `ReplaceWithSurj`, `Fibre` and `Desuspension`.

4.3. The ProjectiveFreeSummand function

We introduce a new algorithm to compute the projective-free summand of a kG -module M , and show that the idea in Section 4.1 can be applied to improve the algorithm. The existing code for computing the projective-free summand first computes the indecomposable summands of M , and then tests each of these summands and excludes the projective ones. This consumes more memory and time. The new algorithm will also need to decompose the regular representation once in order to find all indecomposable projective kG -modules, but it appears to be significantly faster than the old one. See the next section for an example that compares the time needed for the different algorithms for computing the projective-free summand.

Let $f_i : P_i \rightarrow M$ be a set of maps that is jointly surjective, with each P_i being indecomposable and projective, and let $f : N \rightarrow M$ be a map to M . Recall that we write $f + f_i$ for the map

$N \oplus P_i \rightarrow M$ that is f on N and f_i on P_i . We can compute the projective-free summand of M by the following algorithm:

```

ProjectiveFreeSummand = function with one input: a module M
  f_i = a set of maps from P_i to M that is jointly surjective, with
        each P_i being indecomposable and projective
  f = zero map from zero module to M
  r = 0
  for each f_i
    newf = f + f_i
    newr = Rank(newf)
    if newr == r + dimension of P_i then
      f = newf
      r = newr
  return quotient module of M by the image of f

```

By construction, the image of f is a summand of M that is projective. On the other hand, if P is an indecomposable projective summand of M , then there exists some f_i such that f_i maps isomorphically onto P . By induction on M/P , one can show that the image of f finally becomes the projective summand of M . The algorithm works for any set of maps $f_i : P_i \rightarrow M$ that is jointly surjective with each P_i being indecomposable and projective. Since the projective modules P_i are required to be indecomposable, we need to call the **Decompose** function here to find them. And this is the only place that we need to use **Decompose**.

There are different ways to get the maps f_i . The intuitive idea will be computing a basis for $\text{Hom}(P_i, M)$ for each indecomposable projective P_i . Or we can use the projective cover of M here, which can be computed by **ReplaceWithSurj**. This idea can reduce the number of rank computations, but we pay the cost of checking more conditions in the loops and doing more matrix multiplications. However, there will be potential savings in time as we apply this idea and avoid the unneeded loops. We have implemented the latter algorithm in GAP and compared it with the existing algorithm. The results will be presented in the next section.

5. EXAMPLES

In this section, we give examples of computations with the new code. We compare the new code with the old code in Section 5.1, and show that the new code is faster in computing suspensions and desuspensions. Then we make computations for the groups Q_8 and A_4 in Section 5.2, providing evidence for the conjectures that the generating number of Q_8 is 3 and that the generating number of A_4 is 2. And in Section 5.3, we make computations for the group $C_3 \times S_3$, where $\text{Thick}\langle k \rangle \neq \text{StMod}(B_0)$.

5.1. Comparing the new code with the old code

As a special example of fibres and cofibres, we begin with an easy computation of suspensions and desuspensions of the trivial representation for the alternating group A_4 over the field $GF(4)$, and compare the time used by the different versions of the functions **Suspension** and **Desuspension**. We iterate **Suspension** or **Desuspension** to compute $\Sigma^n k$ and measure the total time used.

$\Sigma^n(k)$	$n = 50$		$n = -50$	
	Dimension	Time	Dimension	Time
new function	101	5.1s	101	5.1s
old function	109	34.8s	109	31.1s

Since the old function adds free summands to the target to replace a map with an injection, and similarly for replacing a map with a surjection, the replacement we get by using the old code can fail to be minimal for non- p -groups, which produces projective summands in the answer. In the example, it raises the dimension of $\Sigma^{\pm 50} k$ by 8. To get the optimal answer using the old function,

there is an extra step to determine the projective-free summand, while we have shown that the new algorithm always produces an optimal answer. It is also clear from the table that the new code is faster than the old code.

Now we compare the time needed for the different versions of **ProjectiveFreeSummand** to compute the projective-free summands of $\Sigma^{30}k$ and $\Sigma^{31}k$. We take smaller modules here to reduce the time for the tests. Note that the dimensions of $\tilde{\Sigma}^{30}k$ and $\tilde{\Sigma}^{31}k$ are 61 and 63, respectively. We pre-compute the modules $\Sigma^{30}k$ and $\Sigma^{31}k$ with the old function, and take the result as our models. The old function returns a module of dimension 61 for $\Sigma^{30}k$, which is actually projective-free. But it returns a module of dimension 71 for $\Sigma^{31}k$, which contains a projective summand of dimension 8. Indeed, the old function first finds a free cover $f : F \rightarrow \Sigma^{30}k$ of Σk and then computes $\Sigma^{31}k$ as $\ker(f)$. In our example, F has dimension 132 and is the minimal free cover of $\Sigma^{30}k$. Hence, a module of dimension 71 is the best answer we can get for $\Sigma^{31}k$ using the old function in this case. The following table shows the time needed in computing the projective summands of the pre-computed modules $\Sigma^{30}k$ and $\Sigma^{31}k$:

ProjectiveFreeSummand	$\Sigma^{30}k$, dimension 61	$\Sigma^{31}k$, dimension 71
	Time	Time
new function	0.11s	0.17s
old function	52.1s	71.0s
Decompose	52.0s	70.8s

Recall that the old **ProjectiveFreeSummand** function first decomposes a module into the sum of its indecomposable summands and then excludes the summands that are projective. The last line in the table shows the time spent to decompose the module in the old method for computing the projective-free summand. It shows that decomposing the module is the dominant part of the old method. Even for the module $\Sigma^{30}k$, which is projective-free, it takes a long time for the computer to check with the old code that it does not contain a projective summand. One also sees clearly from the table that the new function for computing the projective-free summand is significantly faster.

For a p -group, since the regular representation is indecomposable, the old function generally produces an optimal answer. But the new **Suspension** function is still faster in this case, as one can see in the following table, where we compute $\Sigma^{\pm 50}k$ for the group $C_3 \times C_3$ over the field $GF(3)$:

$\Sigma^n(k)$	$n = 50$		$n = -50$	
	Dimension	Time	Dimension	Time
new function	226	19.2s	226	19.9s
old function	226	147.2s	226	117.5s

Note that it is not guaranteed by the old algorithm that the answer is going to be optimal, even for a p -group. Also note that it takes more time for the old function to compute $\Sigma^{50}k$ than to compute $\Sigma^{-50}k$ because the old code needs more time to find an injection from a module M into a free module in order to compute ΣM .

5.2. Computations in C_9 , Q_8 , and A_4

We test our code for the cyclic group C_9 of order 9 with $k = GF(3)$, the quaternion group Q_8 of order 8 with $k = GF(2)$, and the alternating group A_4 of order 12 with $k = GF(4)$. Note that the cohomology of C_9 has periodicity 2 and that the cohomology of Q_8 has periodicity 4, so we can compute the generating lengths of kC_9 and kQ_8 -modules exactly in these cases. Recall that the generating number of kC_9 is 4, the generating number of kQ_8 is 3 or 4, and the generating number of kA_4 is 2, 3 or 4 [6, 10, 11]. In the following examples, we will create modules using the function **CreateRandomModule** introduced in Section 4.2, and keep the cofibres R_n with $n \geq 3$, so that R_n can have lengths greater than or equal to 4. Then we compute their generating lengths.

For the group C_9 , we first record the dimensions and lengths of R_3 and R_4 . We performed 6 trials and get

n	3	4	3	4	3	4	3	4	3	4	3	4
Dimension	17	22	30	29	17	8	22	15	7	15	7	16
Length	1	2	2	3	1	1	2	2	3	4	2	2

The process seldom produces a module that achieves that generating number 4. But when we take larger n , we find see more kC_9 -modules of length 4:

n	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Dimension	22	14	20	19	11	11	11	12	11	19	18	18	8	16	9
Length	2	2	3	4	4	4	4	4	3	3	3	3	3	3	2

It is interesting to note that the lengths can decrease in a single trial as we take more steps to build up the modules. Now we repeat the trial many times and check the number of appearances of the modules of different generating lengths. The following table is the result we get from a total of 100 trials:

n	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Length = 1	72	45	30	26	22	14	12	15	11	10	10	10	9	13	12	10
Length = 2	28	44	47	34	35	36	38	30	29	29	28	26	25	21	20	23
Length = 3	0	11	19	29	22	22	22	25	31	29	30	28	31	24	28	32
Length = 4	0	0	4	11	21	28	28	30	29	32	32	36	35	42	40	35

We can see that, for $n = 2$, we only get modules of lengths less than or equal to 2, and similarly for $n = 3$. As n grows larger, we start to see modules of greater lengths, and the distribution of modules of different lengths becomes quite steady for $n \geq 10$, which resembles the behaviour of a Markov chain. We also see that the modules of top lengths appear at a quite high frequency.

We have performed many more trials for C_9 and see this pattern show up again. But this is a very special example with the group being a cyclic p -group. In general, it is an interesting question to see whether there is a similar pattern for any finite group.

Now we apply the method to study kQ_8 -modules. In this case, we are looking for a kQ_8 -module of length 4. It would imply that the generating number of kQ_8 is 4. We have tried to build up kQ_8 -modules with n up to 100, but in all the examples, there are no kQ_8 -modules of length 4, strongly suggesting that the generating number of kQ_8 is 3.

Conjecture 5.1. *Let $G = Q_8$ and k be a field of characteristic 2. Then*

$$\text{generating number of } kQ_8 = 3.$$

For evidence, here is the result when we built up kQ_8 -modules with $n = 10$. We allowed up to 5 summands in each step to build up the modules and performed a total of 200 trials:

n	4	5	6	7	8	9	10
Length = 1	3	1	1	1	2	0	1
Length = 2	46	36	26	28	19	24	25
Length = 3	151	163	173	171	179	176	174

We have not included the line with Length = 4, since we never encountered a kQ_8 -module with generating length 4, which would have disproved the conjecture.

Similarly, we have built up kA_4 -modules with $n = 10$ and up to 5 summands. The modules all have length 2, making us believe that the generating number of kA_4 is 2.

Conjecture 5.2. *Let $G = A_4$ and k be a field of characteristic 2. Then*

$$\text{generating number of } kA_4 = 2.$$

5.3. The group $C_3 \times S_3$ at the prime 3

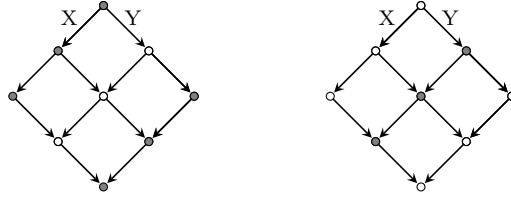
We know from [11, Theorem 4.7] that if the thick subcategory $\text{Thick}\langle k \rangle$ generated by the trivial representation k in $\text{stmod}(kG)$ consists of all the modules in the principal block, that is,

$$\text{Thick}\langle k \rangle = \text{stmod}(B_0), \quad (5.1)$$

then the ghost number of the group algebra kG is finite. In general, when condition 5.1 fails, we don't know whether the ghost number of kG is finite or not. In this case, we can show that a module M is in $\text{Thick}\langle k \rangle$ by showing that it has finite generating length. We make computations for the group $C_3 \times S_3$ in this section, where condition 5.1 fails.

Let $G = C_3 \times S_3$ be the direct product of the cyclic group C_3 of order three and the symmetric group S_3 on three letters. Let k be a field of characteristic 3. We write x for a generator of C_3 , $y = (123)$ for an element of order 3 in S_3 and $z = (12)$ for an element of order 2 in S_3 . Thus G is a group on three generators x , y , and z subject to the relations $x^3 = y^3 = z^2 = 1$, $xy = yx$, $xz = zx$, and $yz = zy^2$.

There are two simple kG -modules k and ϵ . Here k is the trivial representation and ϵ is a 1-dimensional module with z acting as -1 . Since the principal idempotent of kG is 1 [14], both k and ϵ are in the principal block. We will show in a moment that the simple module ϵ is not in $\text{Thick}\langle k \rangle$, hence $\text{Thick}\langle k \rangle \neq \text{stmod}(B_0)$. By Lemma 4.2, the modules k and ϵ correspond to the indecomposable projective modules sketched below:



Here we use a solid dot for k and a circle for ϵ . The arrows down-left indicate the action of $X = 1 - x$, and the arrows down-right indicate the action of $Y = y - y^2$. Note that $Xz = zX$ and $Yz = -zY$.

With an abuse of notation, we write ϵ for both of its restrictions to $C_3 \times C_2$ and S_3 . Restricting to $C_3 \times C_2$, one easily sees that ϵ is not in the principal block of $k(C_3 \times C_2)$, hence cannot be in $\text{Thick}_{C_3 \times C_2}\langle k \rangle$. Since the restriction functor is triangulated, it follows that ϵ is not in $\text{Thick}_G\langle k \rangle$.

More generally, we know that there are only 6 indecomposable $k(C_3 \times C_2)$ -modules:



Again we use a solid dot for k and a circle for ϵ , and the arrows downward indicate the action of $X = 1 - x$. It is clear that the first three modules are in $\text{Thick}_{C_3 \times C_2}\langle k \rangle$. We know that ϵ is not in $\text{Thick}_{C_3 \times C_2}\langle k \rangle$, and the fifth module is isomorphic to $\Omega\epsilon$ in $\text{stmod}(k(C_3 \times C_2))$, hence is not in $\text{Thick}_{C_3 \times C_2}\langle k \rangle$ either. The last module is projective as a $k(C_3 \times C_2)$ -module, hence is in $\text{Thick}_{C_3 \times C_2}\langle k \rangle$. Now we can deduce the following proposition.

Proposition 5.3. *Let $G = C_3 \times S_3$ and let k be a field of characteristic 3. Let M be a kG -module. If M is in $\text{Thick}\langle k \rangle$, then the modules*

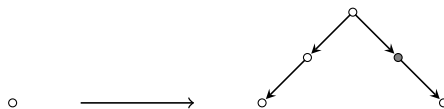


cannot be summands of $M \downarrow_{C_3 \times C_2}$.

□

Conversely, we can view the $k(C_3 \times C_2)$ -modules as kG -modules with trivial y -action. Again, it is easy to see that the first three modules listed above are in $\text{Thick}_G\langle k \rangle$. One also sees that the three-dimensional modules in the list are induced up from the subgroup S_3 , as $k \uparrow^G$ and $\epsilon \uparrow^G$. Since $\Omega^2 k \cong \epsilon$ in $\text{stmod}(kS_3)$, the last module $\epsilon \uparrow^G$ is a double suspension of the third one $k \uparrow^G$ in $\text{stmod}(kG)$, hence is in $\text{Thick}_G\langle k \rangle$ too. But the other two modules are not in $\text{Thick}_G\langle k \rangle$ by Proposition 5.3. We conjecture that the converse of the proposition is also true. In the following example, we construct a module M that satisfies the condition in Proposition 5.3 and show that it is in $\text{Thick}_G\langle k \rangle$. Indeed, this is equivalent to showing that the generating length of M is finite by Lemma 3.3.

Example 5.4. We consider the cokernel M of the non-zero map f



that sends ϵ to the difference of the bottom elements. By Proposition 5.3, the domain and codomain of f are not in $\text{Thick}_G\langle k \rangle$. But $M \downarrow_{C_3 \times C_2}$ is in $\text{Thick}_{C_3 \times C_2}\langle k \rangle$. We can compute the generating length of M (more precisely, an upper bound of the generating length of M) with the **Length** function, and show that

$$M \text{ is in } \text{Thick}_G\langle k \rangle.$$

The **Length** function tells us that $\text{gel}_3(M) = 3$, and it follows that $\text{gel}(M) \leq 3$. Now we actually show that $\text{gel}(M) = 3$. To compute the lower bound, we consider left multiplication by the central element $1 - x$ on M . Restricting to $C_3 \times C_3$, we know that $1 - x$ is a ghost and $(1 - x)^2$ is stably non-trivial. Then, by Theorem 3.2 in [11], $1 - x$ is a simple ghost, hence a ghost, on M . Since the restriction functor to the Sylow p -subgroup is faithful, the generating length of M is at least 3.

REFERENCES

- [1] D. J. Benson. *Representations and cohomology I*. Cambridge Univ. Press, Cambridge, 1998.
- [2] D. J. Benson, S. K. Chebolu, J. D. Christensen and J. Mináč. The generating hypothesis for the stable module category of a p -group. *Journal of Algebra* **310**(1) (2007), 428–433.
- [3] A. Bondal and M. Van den Bergh. Generators and representability of functors in commutative and noncommutative geometry. *Moscow Math. J.* **3** (2003), 1–36.
- [4] J. F. Carlson, S. K. Chebolu and J. Mináč. Freyd’s generating hypothesis with almost split sequences. *Proc. Amer. Math. Soc.* **137** (2009), 2575–2580.
- [5] S. K. Chebolu, J. D. Christensen and J. Mináč. Groups which do not admit ghosts. *Proc. Amer. Math. Soc.* **136**(4) (2008), 1171–1179.
- [6] S. K. Chebolu, J. D. Christensen and J. Mináč. Ghosts in modular representation theory. *Advances in Mathematics* **217**(6) (2008), 2782–2799.
- [7] S. K. Chebolu, J. D. Christensen and J. Mináč. Freyd’s generating hypothesis for groups with periodic cohomology. *Canadian Mathematical Bulletin*, **55**(1) (2012), 48–59.
- [8] J. D. Christensen. Ideals in triangulated categories: phantoms, ghosts and skeleta. *Adv. Math.* **136**(2) (1998), 284–339.
- [9] J. D. Christensen, Brad Froehle, Robert Hank, Roland Loetscher, Bryan Simpkins, Peter Webb, and others. *GAP package ‘reps’*. <http://www.math.umn.edu/~webb/GAPfiles/>.
- [10] J. D. Christensen and G. Wang. Ghost numbers of group algebras. *Algebras and Representation Theory*, **18**(1) (2015), 1–33.
- [11] J. D. Christensen and G. Wang. Ghost numbers of group algebras II. *Algebras and Representation Theory*, (2015). doi:10.1007/s10468-015-9519-x
- [12] P. Freyd. Stable homotopy. In *Proc. Conf. Categorical Algebra (La Jolla, Calif., 1965)*, pages 121–172. Springer, New York, 1966.
- [13] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.7.5*; 2014, <http://www.gap-system.org>.
- [14] B. Külshammer. The principal block idempotent. *Arch. Math.* **56**(4) (1991), 313–319.
- [15] G. Wang. *GAP package ‘reps-code’*. http://www-home.math.uwo.ca/~gwang72/reps_code/reps-code.

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF WESTERN ONTARIO, LONDON, ON N6A 5B7, CANADA
E-mail address: gwang72@uwo.ca